9 April 2021

Three-Dimensional Matching
and Subset Sum

---

Announcement: Check pinned Piazza post for Prelim 2 review materials.

---

k = 2:

1st elt. of JS
2nd elt. of JS

---

Recap: 3DM is to decide if the 3n elements can be partitioned into disjoint 3-element sets, each of which is among the given ones.
Q1. How to express the "don't pick both endpoints of an edge" constraint using 3DM?

Q2. What to do with leftover blue nodes?

Trouble: if a vertex has $> 1$ neighbor, it becomes impossible to represent the edges of the graph accurately.
Solution: use a "pinwheel gadget" for each vertex.
If the vertex belongs to $d$ edges, the pinwheel will have:
- $d$ red nodes
- $d$ green nodes
- $2d$ blue nodes
($d$ of them correspond to "choosing" the vertex and will be shared with pinwheels of neighboring vertices.)
Do this for each vertex.
There are 2 ways to cover the red-green nodes in a vertex gadget.

1. Connect each red to clockwise green neighbor. Cover all the blue nodes on the edge leaving that vertex.

"Choosing the vertex to belong to the independent set."

2. Connect to counterclockwise neighbor. Covers all the internal blue nodes.

Enforce "choose k vertices" constraint by having k additional red-green pairs each connected to one internal blue node of each plucked.

\[ \text{Total: } k + \sum_{v \in V} d(v) \text{ red} \]
Property: There exists a 3D matching that covers all red & green nodes (not necessarily all blue).

$\iff G$ has a k-element independent set.

Summary: We showed a problem is NP-Complete but it's not 3D matching. It is "3D Matching With leftover blue nodes" (3DMWLB).

Given an input instance of 3DM, can we find disjoint 3-element sets among the given ones that cover all red and green nodes?

Finally, show 3DMWLB $\leq_p$ 3DM:

Given input instance of 3DM, create a 3DM instance with sets $R, G, B, \ |R|=n, |G|=n, |B|=n+s \ (s \geq 0)$.

Create $R', G', B$:

$R' = R \cup R_1, |R'| = 5$

$G' = G \cup G_1, |G'| = 5$
and the 3-element sets in this 3DM
instance are all the sets given
in the original 3DMWLB instance,
and \(s^2(n+5)\) additional sets:
each set formed from one
element of \(P_i\), one elt of \(G_i\),
one element of \(B\).

We’ve shown \(\textsc{IND-SET} \subseteq \textsc{3DMWLB} \subseteq \textsc{3DM}\).

\underline{SUBSET SUM}. Given positive integers
\(w_1, w_2, \ldots, w_n\) and target sum \(W\)
is there a subset of \(\{w_1, \ldots, w_n\}\) that
sums up to \(W\)?

\underline{KNAPSACK}. Given items with integer weights and
value \((w_i, v_i)\), and budget \(B\) and
target value \(V\), is there a subset
with combined weight \(\leq B\) and
combined value \(\geq V\)?

\underline{SUBSET SUM} \(\leq_p \underline{KNAPSACK}\)

\(w_i \rightarrow \text{item with } V = w_i\)

\(\text{target } W \rightarrow B = V = W\).
Recall the dynamic program for knapsack run in \( O(nW) \) time, which is pseudopolynomial but not polynomial.

If \( W \) has \( m \) binary digits, then the knapsack problem has input size \( O(nm) \) but the dynam program takes \( O(n \cdot 2^m) \) time to solve it.

Shawing SUBSET SUM is NP-complete.
Reduce 3DM \( \leq_p \) SUBSET SUM.